Automatic Reconstruction of Spacecraft 3D Shape from Imagery

Conrad Poelman, Ph.D.
Robert Radtke
Harry Voorhees
Stellar Science Ltd. Co.

ABSTRACT

We describe a system that computes the three-dimensional (3D) shape of a spacecraft from a sequence of uncalibrated, two-dimensional images. While the mathematics of multi-view geometry is well understood, building a system that accurately recovers 3D shape from real imagery remains an art. A novel aspect of our approach is the combination of algorithms from computer vision, photogrammetry, and computer graphics. We demonstrate our system by computing spacecraft models from imagery taken by the Air Force Research Laboratory's XSS-10 satellite and DARPA's Orbital Express satellite.

Using feature tie points (each identified in two or more images), we compute the relative motion of each frame and the 3D location of each feature using iterative linear factorization followed by non-linear bundle adjustment. The "point cloud" that results from this traditional shape-from-motion approach is typically too sparse to generate a detailed 3D model. Therefore, we use the computed motion solution as input to a volumetric silhouette-carving algorithm, which constructs a solid 3D model based on viewpoint consistency with the image frames. The resulting voxel model is then converted to a facet-based surface representation and is texture-mapped, yielding realistic images from arbitrary viewpoints. We also illustrate other applications of the algorithm, including 3D mensuration and stereoscopic 3D movie generation.

1. MOTIVATION

This research investigates the feasibility of applying computer vision "structure from motion" algorithms to the domain of space situational awareness (SSA). The research goal is to automatically compute the three-dimensional (3D) shape, attitude, and motion of a satellite from a sequence of uncalibrated images of it. This capability could aid SSA by automatically processing imagery taken from ground-based telescopes and alerting operators to any unexpected changes in spacecraft configuration. The automatic recovery of 3D shape from imagery has other applications as well, including object identification, generating virtual "walk-throughs" of buildings and urban environments, and autonomous robot navigation.

2. BACKGROUND

As an object moves relative to a camera or telescope, the sequence of images captured by the camera reveals fundamental geometric shape information about the object or scene (see Fig. 1). Nearer parts of the object move in the image differently than distant parts, portions of the object may become hidden by other parts, the intensity of reflected light changes as the viewing angle changes, and so forth. The task of recovering the shape of the object, generally without specific knowledge of the relative motion, is referred to as the "structure from motion" (SFM) problem.

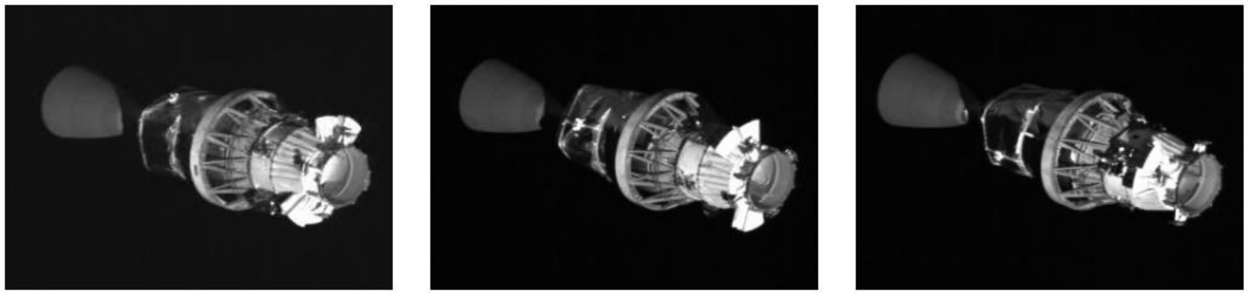


Fig. 1: A subset of the images of a rocket body taken from the XSS-10 satellite. The full sequence has 25 images and depicts over 720 degrees of rotation primarily about the rocket body's main axis.

The SFM problem has been studied since the early 1980s. While mathematical techniques for computing 3D geometry from multiple, uncalibrated cameras are well documented [1], there remain a number of issues that make implementing a 3D software system tricky in practice:

- accurately locating and tracking object features whose appearance can vary widely with differences in viewing angle and illumination, and which may disappear from view
- discounting erroneous feature tracks
- handling multiple moving parts
- transforming a 3D "point cloud" into a more rich model of shape.

The purpose of this project is to address these problems, specifically as applied to imagery of on-orbit satellites.

3. ALGORITHM OVERVIEW

Our 3D shape and motion reconstruction process combines a number of algorithms from computer vision and graphics, and is depicted in Fig. 2. The process can be organized into two main stages: first, computing a sparse representation of shape along with the object motion relative to the camera, and second, computing a more detailed, rich shape representation. In the next few sections, we describe the computation of each shape representation and some applications of each.

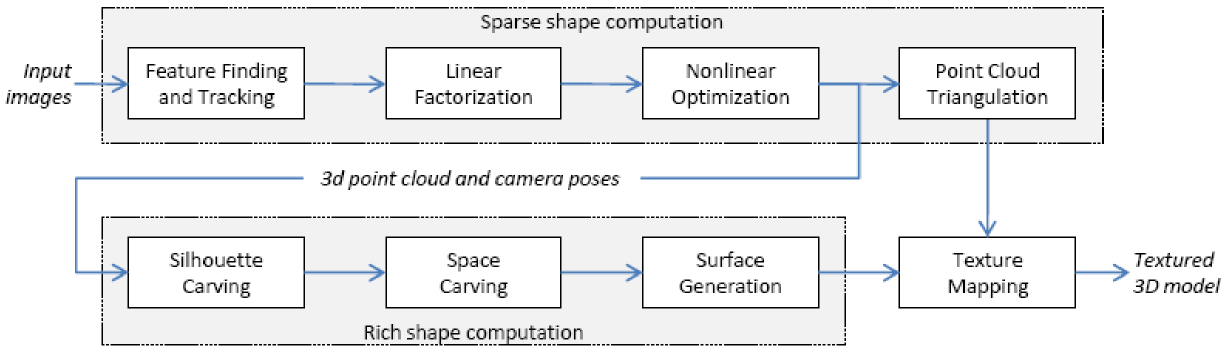


Fig. 2. Overview of 3D shape reconstruction algorithm

4. COMPUTING SPARSE SHAPE

4.1 Feature Finding and Tracking

The "structure from motion" approach is based on tracking the projected location of features on a rigid 3D object among multiple images taken from different viewpoints, as shown in Fig. 3. In the examples shown here, features are located and matched in each frame by a person. We have implemented a number of techniques described in the computer vision literature [2,3,4] as well as novel approaches to automatically detect and match or track features. To date these techniques have had some success, but the fact that the appearance of a physical point on the object changes significantly as the image rotates relative to the camera and light source, combined with the often large displacements from one image to the next, have limited our ability to automatically and precisely track physical points on the object. Therefore manual intervention was applied to feature finding and matching in the sequences illustrated in this report.

4.2 Linear Factorization

In the point-based SFM problem, we are given the image point locations, called "observations", of n different object feature points in m different image frames, which we denote as $o_j^i = [u_j^i \; v_j^i]^T$, where i ranges from 1 to m and j ranges from 1 to n. We denote each object point's unknown 3D location as $\mathbf{s}_j = [x_j \; y_j \; z_j]^\mathrm{T}$, and denote the camera's unknown position and orientation as the 2×4 affine transformation matrix $M^i = [s_{1,1}^i \; s_{1,2}^i \; s_{1,3}^i \; t_x^i; \; s_{2,1}^i \; s_{2,2}^i \; s_{2,3}^i \; t_y^i]$.

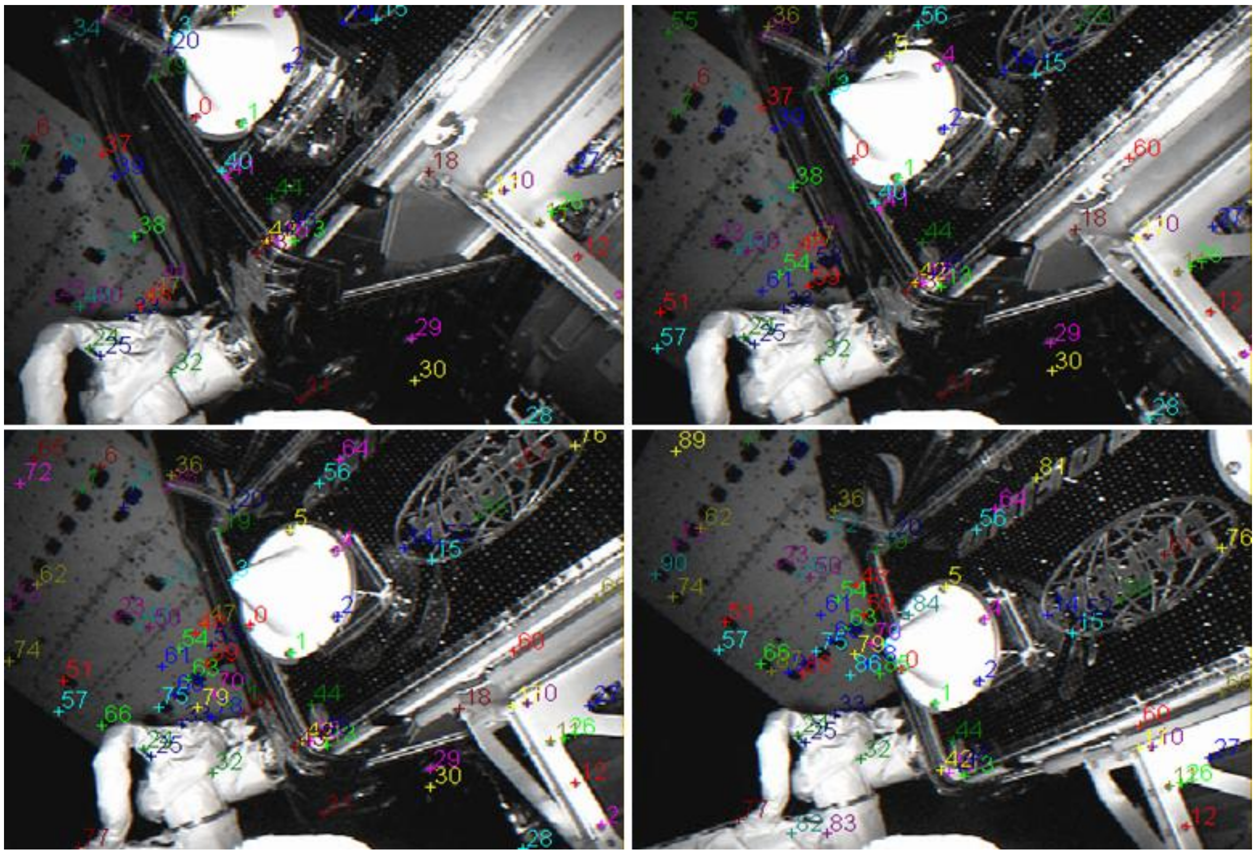


Fig. 3: Manually specified feature tracks for four frames of the Orbital Express test scenario

When the object is relatively far from the camera, perspective effects can be approximated with scaled orthographic or paraperspective projection models, and the projection of feature *j* onto image *i* as pictured in Fig. 4 can be modeled by the equation $\boldsymbol{o}_{ij} = M^i \boldsymbol{s}_j$ [5,6].

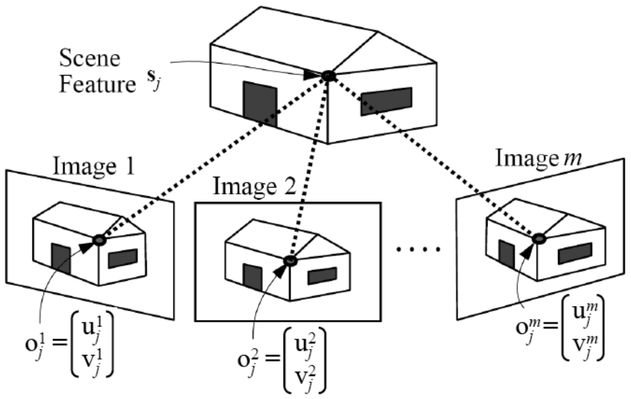

Scene Feature $\mathbf{s}_j$
Image 1
Image 2
Image m
$o_j^1 = \begin{pmatrix} u_j^1 \\ v_j^1 \end{pmatrix}$
$o_j^2 = \begin{pmatrix} u_j^2 \\ v_j^2 \end{pmatrix}$
$o_j^m = \begin{pmatrix} u_j^m \\ v_j^m \end{pmatrix}$

Fig. 4. An object feature with corresponding image features (adapted from [5])

We can formulate these equations, for all *n* points and *m* image frames, into a single linear system,

$$\begin{pmatrix} u_1^1 & u_2^1 & . & . & . & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \\ . & . & & & & . \\ . & . & & & & . \\ . & . & & & & . \\ u_1^m & u_2^m & & & & u_n^m \\ v_1^m & v_2^m & . & . & . & v_n^m \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \\ & . & & \\ & . & & \\ & . & & \\ s_{1,1}^m & s_{1,2}^m & s_{1,3}^m & t_x^m \\ s_{2,1}^m & s_{2,2}^m & s_{2,3}^m & t_y^m \end{pmatrix} \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix} \tag{1}$$

or, using matrix notation, $O = MS$. Matrix O contains the known parameters (the observed feature locations), matrix M contains the unknown "motion" parameters (camera pose and projection parameters), and matrix S contains the unknown 3D "shape" parameters. For large numbers of points and images, there are more known parameters than unknown, and the best-fit least-squares solution can be computed robustly and efficiently using the "factorization method" [2]. This method uses singular value decomposition (SVD) to yield estimates of the shape and motion, $\hat{S}$ and $\widehat{M}$, which are specified up to an unknown affine transformation.

In practice, it is unreasonable to require that every feature be visible in every frame. As the object rotates in 3D space relative to the camera, features may disappear from view due to self-occlusion, and new features may appear in subsequent frames. This yields a sparse observation matrix, and SVD cannot be applied to matrices with missing values. To handle this case, our algorithm finds columns and rows that form the largest dense submatrix and applies SVD to it, yielding the shape and motion values of the corresponding points and frames. Then the remaining rows and columns are iteratively folded into the solution to compute the remaining values of $\hat{S}$ and $\widehat{M}$.

Finally the actual shape and motion are computed from the affine estimates of shape and motion by imposing normalization constraints on the camera projection matrix. Rather than being general affine transforms, which can introduce stretching and skewing effects, we incorporate the knowledge that the camera's horizontal and vertical projection axes are orthogonal to each other and equally scaled, or scaled by a known aspect ratio.

While the factorization method is quite robust, it should be noted that the computed solution may be imperfect for a number of reasons: First, some feature tracks may be incorrect. Second, errors may accumulate during the iterative process of adding rows and columns to the solution. Third, converting affine motion to a rigid-body motion may have introduced errors. And fourth, if the camera is near the object, then scaled orthography may not be an accurate projection model. Therefore we refine these shape and motion estimates with a non-linear optimization step, described next.

4.3 Nonlinear Optimization

Like the "bundle adjustment" technique from photogrammetry, the nonlinear shape and motion optimizer attempts to find an optimal shape and motion solution for the given data. We minimize the sum-of-squares error between the original observations and the projections of the computed features $\chi^2 = \sum_{ij} \left\| \boldsymbol{o}_{ij} - \hat{\boldsymbol{o}}_{ij} \right\|^2$. Here $\boldsymbol{o}_{ij}$ are the observed feature locations, and $\hat{\boldsymbol{o}}_{ij}$ are the feature locations predicted by the current shape and motion solution ($\hat{S}$ and $\widehat{M}$), where a more general non-linear camera model now can be used to correctly model perspective effects.

Our nonlinear shape and motion optimizer is based on the Levenberg-Marquardt optimization algorithm that involves repeatedly solving a linear matrix equation $Hx = b$ involving the Hessian matrix H. When large numbers of feature points and/or large numbers of images are used, the problem size becomes very large. Each feature point adds three unknowns to the system of equations (the x, y, and z locations of the point), and each image frame adds six unknowns to the system (the x, y, and z location of the camera as well as the roll, pitch, and yaw that describe the camera orientation). Thus a problem that has n feature points in m image frames requires solving for a total of $u = 3n + 6m$ unknowns, and each step of the iterative optimization requires the solution of a $u \times u$ matrix equation.

Fortunately, we can take advantage of the sparse, symmetric form of matrix H to solve the large matrix equation efficiently. The matrix is diagonally dominant, meaning that the majority of the large values lie in small blocks laid out along the diagonal of the matrix, and most other elements of the matrix are zero or very small. This layout lends itself to faster inversion using the biconjugate gradient method [7].

4.4 Outlier Rejection

Before refining the solution, as described above, it may be necessary to discount outliers, i.e., poorly tracked features, which would corrupt the solution. To do this, we identify the observation (*i,j*) whose removal would result in the greatest reduction of χ^2error. If the reduction is significant, we remove that observation from consideration, and look for the next outlier, repeating the process until the error is relatively stable, or a maximum number of outliers have been removed.

To test each observation for potential removal, we re-solve for $\hat{S}$ and $\hat{M}$ without including the candidate outlier. Experimentation has shown that we cannot assume that the observation with the largest individual error contribution $\left\|\boldsymbol{o}_{ij} - \hat{\boldsymbol{o}}_{ij}\right\|^2$ is the outlier, which would be a much quicker test, because adding an outlier to a data set can bias the solution to fit the outlier better than another (inlying) observation.

4.5 Track Partitioning

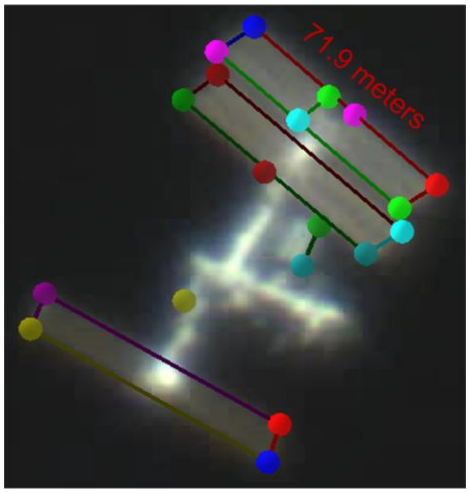
Equation 1 assumes that the all features lie on a single, rigid object. If the object has moving parts whose articulation varies among the images being processed, such as moving solar panels, then we must solve for the shape and motion of each such part independently. While components may not typically move significantly during a single image collection, such motion is common when imagery collections acquired at different times are combined to build the model. We have demonstrated a technique applied to synthetic data that finds such parts by using the Random Sample Consensus (RANSAC) algorithm [4] to find subsets of feature tracks that are consistent with rigid body motion. However, such techniques were not required for any of the image sequences analyzed here.

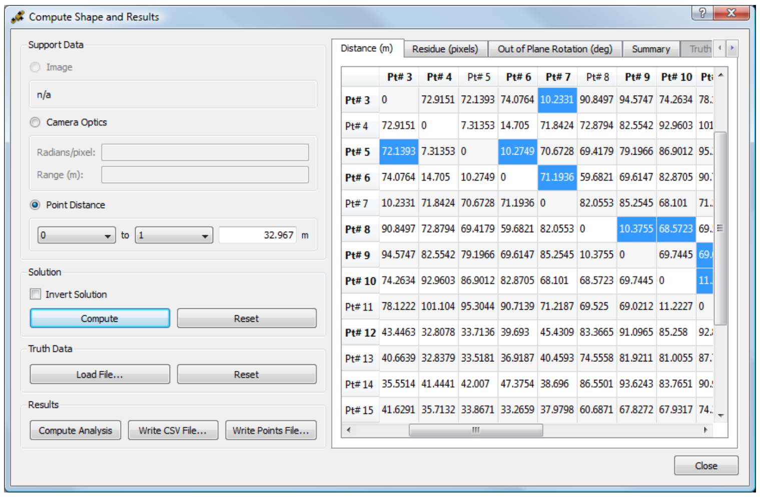
5. APPLICATIONS OF SPARSE SHAPE MODELS

The shape solution $\hat{S}$ computed by the linear and nonlinear solve steps is a 3D "point cloud" that does not specify the surface shape between the feature points. Despite its sparseness, the point cloud is useful for certain applications as described below.

5.1 3D Mensuration

Mensuration can be performed using the point cloud model, particularly if features are defined at the ends of items to be measured. We have developed a 3D mensuration tool called Caliper that uses this approach to obtain accurate measurement from real imagery, using user-defined feature points. Because SFM techniques can only compute shape up to a relative scale factor (images alone cannot distinguish between a house and a scale model of a house), the tool must obtain one additional piece of information in order to properly scale the points. The scale factor may be automatically determined from image metadata, or it can be determined from a single known measurement provided by the user. A mensuration example is shown in Fig. 5. The grid on the right shows the distances between every pair of points that are superimposed on the image on the left.





n/a

0 to 1 32.967 m

Invert Solution

	Pt# 3	Pt# 4	Pt# 5	Pt# 6	Pt# 7	Pt# 8	Pt# 9	Pt# 10	Pt
Pt# 3	0	72.9151	72.1393	74.0764	10.2331	90.8497	94.5747	74.2634	78.
Pt# 4	72.9151	0	7.31353	14.705	71.8424	72.8794	82.5542	92.9603	101
Pt# 5	72.1393	7.31353	0	10.2749	70.6728	69.4179	79.1966	86.9012	95.
Pt# 6	74.0764	14.705	10.2749	0	71.1936	59.6821	69.6147	82.8705	90.
Pt# 7	10.2331	71.8424	70.6728	71.1936	0	82.0553	85.2545	68.101	71.
Pt# 8	90.8497	72.8794	69.4179	59.6821	82.0553	0	10.3755	68.5723	69.
Pt# 9	94.5747	82.5542	79.1966	69.6147	85.2545	10.3755	0	69.7445	69.
Pt# 10	74.2634	92.9603	86.9012	82.8705	68.101	68.5723	69.7445	0	11.
Pt# 11	78.1222	101.104	95.3044	90.7139	71.2187	69.525	69.0212	11.2227	0
Pt# 12	43.4463	32.8078	33.7136	39.693	45.4309	83.3665	91.0965	85.258	92.
Pt# 13	40.6639	32.8379	33.5181	36.9187	40.4593	74.5558	81.9211	81.0055	87.
Pt# 14	35.5514	41.4441	42.007	47.3754	38.696	86.5501	93.6243	83.7651	90.
Pt# 15	41.6291	35.7132	33.8671	33.2659	37.9798	60.6871	67.8272	67.9317	74.

Fig. 5. Image with computed point cloud (left) over which measurements are computed (right)

5.2 Coarse Textured Models

If only a coarse description of shape is needed, then the points can be used to define a triangulated surface model, as shown in Fig. 6. Corresponding triangles of the input imagery can be used to texture map the facets (as described in Section 6.4), yielding a textured, coarse 3D model. While such a model may not be an accurate representation of shape due to the sparseness of feature points, it is often adequate for rendering animations or visualization.

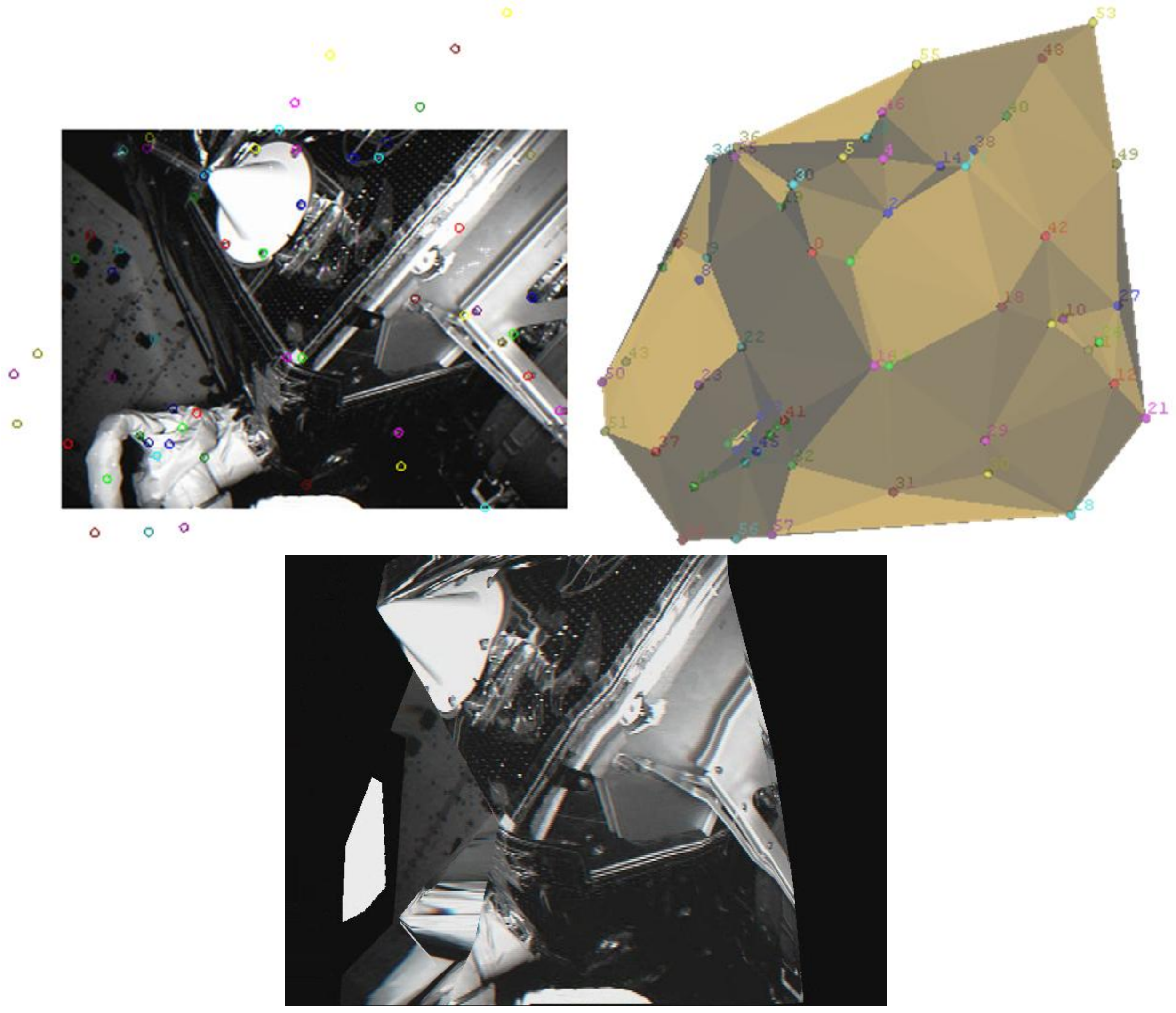


Fig. 6: Triangulated 3D surface (upper right) generated from the point cloud, which is also shown over one of the images used to create it (upper left). The texture mapped 3D surface displayed at a novel angle (lower image).

6. COMPUTING RICH SHAPE

In order to compute a more detailed and realistic representation of 3D shape, we employ a pair of related computer vision techniques known as silhouette carving and space carving. The carving algorithms require an accurate measure of the camera position and orientation in each frame, so the motion computed by the nonlinear solver is a required input for carving, while the sparse shape computed by the previous steps will be replaced with the richer shape.

6.1 Silhouette Carving

The first technique, silhouette carving, is based on the principle that the 3D shape of an object must be consistent with the silhouettes of it imaged by each camera. Silhouette carving requires segmenting each input image into foreground and background. Although the segmentation can be computed automatically in certain cases [8] due the dark background of space, space surveillance images sometimes contain a “halo” around space objects, causing their boundaries to be indistinct. In such cases user assistance is needed to define an accurate silhouette. Once the silhouettes are defined, the silhouette carving automatically computes a discretized, solid model of the 3D shape.

A 3D volumetric voxel model of the shape is initialized to a cube large enough to fill each image. The model is projected onto the image plane according to the computed camera motion (position and orientation) parameters. Voxels in the model that project to the image background are removed. After processing the first image, the model will resemble an extrusion of the image silhouette. The process is repeated for each image, with the model positioned and rotated according to the corresponding camera pose. At the end of the process, the shape model is the most convex shape that is consistent with each image silhouette.

6.2 Space Carving

The second technique, space carving, is used to carve away concavities of the shape and to color the voxels [9,10]. It is based on the assumption that the color of a point on the surface should remain relatively consistent from different viewpoints. The algorithm computes the color of each voxel as predicted by each image in which that voxel is visible. If the measures are inconsistent, it is presumed that the rays that project those colors onto the image originate from different surface points, so the voxel is removed. For example, in Fig. 7 the voxel under consideration appears to cameras 1 and 2 to have different colors because the cameras are imaging different surface points at greater depths than the voxel; therefore, the voxel will be removed from the shape model. When computing voxel color, the algorithm must disregard viewpoints where the voxel under consideration is occluded by other parts of the model, as is the case with camera 3 in the figure, and care must be taken to choose an appropriate order for processing voxels [11]. While the technique works best on color imagery, for monochrome imagery a texture measure can be used instead.

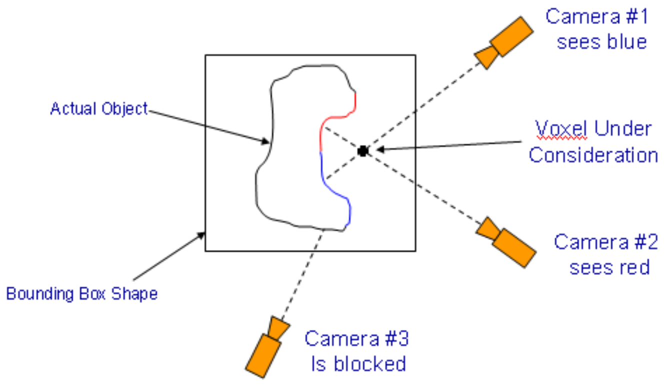


Fig. 7. Color consistency example

With high quality imagery from different views, a fairly detailed volumetric model of shape can be constructed, as shown in Fig. 8.

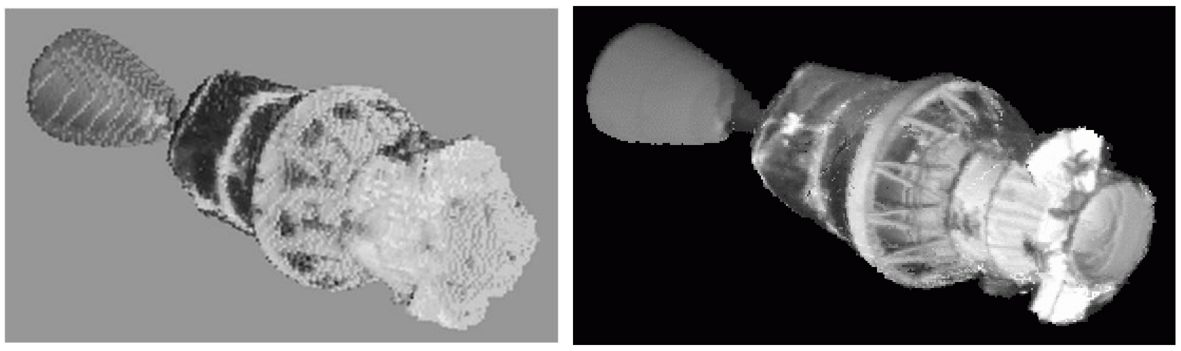


Fig. 8. Computed voxel model of rocket body (left), shown with image-based texture (right)

6.3 Surface Generation

The output of space carving is a three-dimensional matrix of voxels that comprise a discretized, solid model of the sensed object. Ultimately, we would like to represent the model using higher-level shape elements that provide a more accurate representation of the object's surfaces, and if possible, involve fewer elements. A first step towards meeting these goals is to transform the voxel model into a faceted representation of the object's surfaces.

We have implemented a transformation known as "alpha shape" computation. An alpha shape is generalization of the convex hull that corresponds to an intuitive notion of shape at varying levels of detail [12]. Given a dense 3D point cloud, the alpha shape is a triangulated surface whose vertexes are a subset of the point cloud, such that a sphere of a specified radius connects adjacent points. In this case, the centers of the exterior voxels (or their vertices) define the dense point cloud that the alpha shape is fit to. This system is able to replace a complex voxel model with a simpler model composed of surface facets, which can then be texture mapped for a more accurate and smoother visual appearance.

6.4 Texture Mapping

The final step of shape processing is to apply texture to the geometric model. We have developed a hardware-accelerated implementation to accomplish this task, and the technique can be applied to any of the model representations previously described: triangulated point clouds, voxels, or alpha-shape facets. The technique can use up to four of the input images as textures and can operate on bit depths of up to 32 bits per pixel. Typically, the system will select input textures in a way that provides good coverage on all sides of the model. Each fragment in the output image is automatically textured based on the input image whose camera direction most closely matches the fragment's surface normal. The implementation correctly handles occlusion and can even handle cases where none of the four images provides good texture information. Provided that the computed 3D shape is accurate, the algorithm will correctly map the proper texture to each image fragment. The algorithm accounts for occlusion and can detect and handle areas of the model where none of the four images provides good texture information. As shown in Fig. 8, texture mapping can improve the visual appearance of a model even when the underlying solid model voxel resolution is relatively low.

7. APPLICATIONS OF RICH SHAPE MODELS

7.1 Novel View Generation

Using our computed rich shape model, we can synthesize views of object at orientations that were not originally imaged. By generating intermediate frames synthetically, we are able to create smoothly varying movies of an object from more coarsely spaced frames. In Fig. 9, the left and right frames are original images, while the center frame is rendered from the computed model of the rocket body, at an orientation halfway between that of the images.

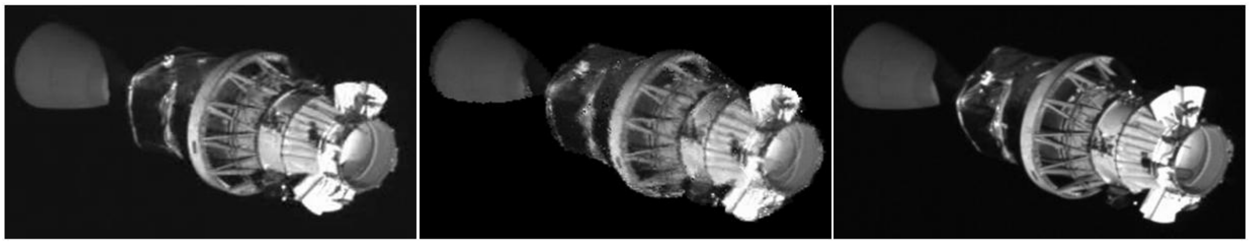


Fig. 9. Two images of the rocket body test sequence (right and left) with a synthetic image rendered using the computed model (center)

It is also possible to render the model at poses that are outside the imaging baseline, as shown in Fig. 10. Note that in the upper rendering, some texture is missing from sides of the model that were not seen in any of the input images. The base of the rocket nozzle is slightly convex, because it was not directly seen; however, its shape is consistent with all of the input images.

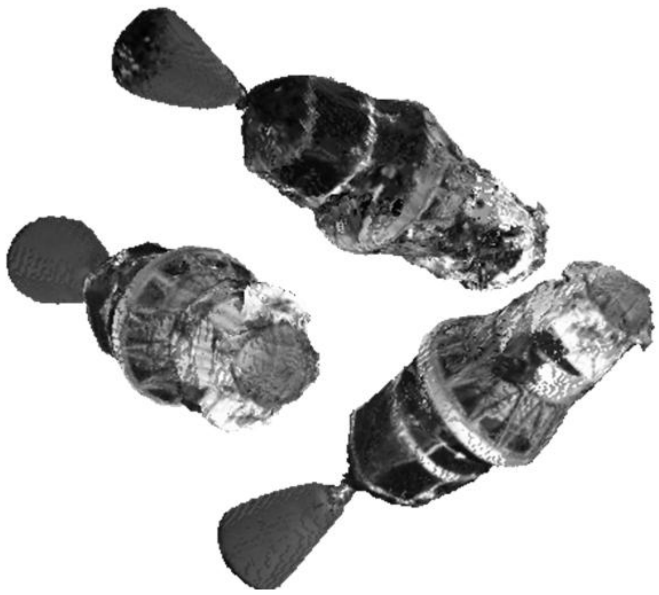


Fig. 10. Rendering of Rocket body at novel orientations

7.2 Stereoscopic Image and Movie Generation

We have developed techniques to create stereoscopic images and movies of 3D models for 3D viewing. Typically, these images are rendered in two colors and are viewable using anaglyph stereo glasses. For a person to perceive depth in a stereo image pair, the object must undergo a small rotation about the vertical axis between the left image and the right. Natural image sequences may not have any two frames with such a rotation, but our generated model can be used to synthesize a second image with exactly the required rotation. Fig. 11 shows a red and cyan stereoscopic image rendered using the computed mesh model of the Orbital Express satellite.

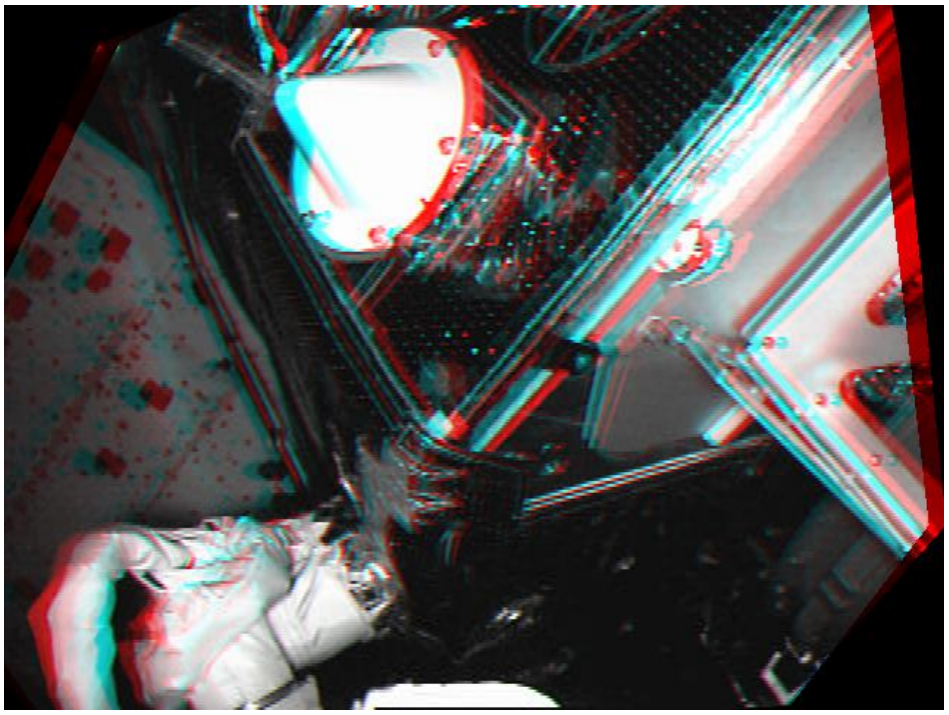


Fig. 11: An anaglyph stereo image created from Orbital Express collection. This image was created by rendering image textures onto a triangulated shape model. Anaglyph glasses are required to achieve a stereo effect.

8. CONCLUSION

We have applied a number of techniques from computer vision and computer graphics to compute 3D models of satellites that are suitable for intermediate frame generation, stereo image visualization, and mensuration. A novel aspect of our approach is the combination of traditional structure-from-motion algorithms with space-carving. This innovation is important for two reasons: First, because we run SFM primarily to compute motion rather than shape, we can accept the relatively small number of features that are detectable in typical SSA imagery. Second, by using the motion solution as an input to silhouette and space carving, we extend the applicability of volumetric carving techniques to uncalibrated imagery. This combination makes it possible to compute a detailed 3D model from uncalibrated imagery with sparse features.

Although we have obtained good results on a number of real data sets, the process is not fully automated, and number of areas could benefit from additional research and development. While we have obtained promising results with automated feature matching and tracking, the examples shown here all required manual intervention to properly identify and match features between images. Other areas that merit further work include automating the background segmentation step of silhouette carving, further development and testing of multi-part finding, and the automatic setting of algorithm parameters.

Despite these limitations, the technology is mature and capable enough to aid in satellite mensuration and model building. Unlike current approaches that use single images for mensuration, the SFM system performs “multi-bundle adjustment” automatically, potentially resulting in more accurate measurements, and enabling analysts to more easily incorporate imagery from multiple views.

REFERENCES

1. Hartley, R. and Zisserman, A. *Multiple View Geometry in Computer Vision*, 2nd ed., Cambridge University Press, 2003.
2. Tomasi, C. and Kanade, T., “Shape and motion from image streams under orthography: A factorization approach,” *International Journal of Computer Vision*, Vol. 6, No. 2, pp.137-154, 1992.
3. Lowe, D. “Distinctive image features from scale-invariant keypoints”, *International Journal of Computer Vision*, Vol. 60, No. 2, pp. 91–110, 2004.
4. Fischler, M. and Bolles, R. “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartograrphy”, *Communications of the ACM*, Vol. 24, No. 6, pp. 381-395, 1981.
5. Kanade, T., and Morris, D. “Factorization methods for structure from motion”, *Philosophical Trans. of the Royal Soc. Of London*, Vo. 356, pp. 1153-1173, 1998.
6. Poelman, C., and Kanade, T. “A paraperspective factorization method for shape and motion recovery”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):206-218, 1997.
7. Barrett, R. et al. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, 2nd ed. Philadelphia, PA: SIAM, 1994 (also http://www.netlib.org/linalg/html_templates/Templates.html).
8. Otsu, N. “A threshold selection method from gray-level histograms”, *IEEE Trans. Systems, Man, and Cybernetics*, Vol. 9, No. 1, pp. 62–66, 1979.
9. Seitz, S. and Dyer, C. “Photorealistic Scene Reconstruction by Voxel Coloring”, *International Journal of Computer Vision*, Vol. 35, No. 2, pp. 151–173, 1999.
10. Kutulakos, K. and Seitz, S. “A theory of shape by space carving”, *International Journal of Computer Vision*, Vol. 38, No. 3, pp. 199–218, 2000.
11. Culbertson, W. B., Malzbender, T. and Slabaugh, G. G., “Generalized Voxel Coloring,” *Proc. International Workshop on Vision Algorithms: Theory and Practice*, pp. 100–115, 1999.
12. Fischer, K. “Introduction to Alpha Shapes”, http://www.inf.ethz.ch/personal/fischerk/pubs/as.pdf.